EXHIBIT 1

EXHIBIT H

AKAMAI'S PRELIMINARY INVALIDITY CONTENTIONS U.S. PATENT NO. 8,750,155[1]

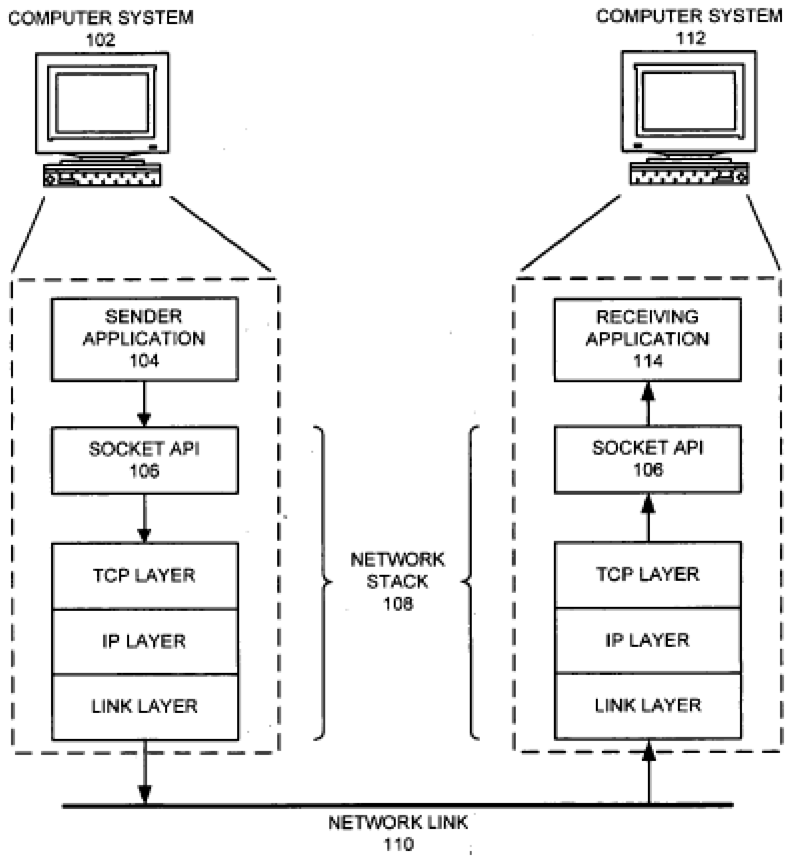
Claims 1-4, 6-8, 10, 11, and 13-20 of the U.S. Patent No. 8,750,155 ("'155 patent") are unpatentable under pre-AIA 35. U.S.C. § 103 in view of U.S. Patent Application Publication No. 2007/0226375 ("Chu")[2] and U.S. Patent Application No. 2007/0156845 ("Devanneaux")[3] as outlined in the chart below.

Asserted Claim	Disclosures in the Prior Art
1. A method for managing delivery of content in a system comprising a server and an end user computer, comprising:	"The transmission control protocol (TCP) is part of the core Internet protocol which is used to transfer data between computing devices. . . . . what is needed are architectures and methods that facilitate congestion control for TCP without the limitations of existing approaches. One embodiment of the present invention provides a plug-in architecture for a network stack in an operating system. The network stack includes a set of functions configured to modify a set of parameters that are likely to change based on the network environment. The architecture includes a plug-in framework within the network stack that allows the set of functions to be dynamically changed in order to change the TCP behavior of the network stack to suit the network environment." Chu at ¶¶ 5-8. Examples of delivery of content from a server to an end user computer are presented in Chu, e.g., at ¶66: "For instance, a connection to a local wireless IP address may need different TCP behavior than a

[1] These preliminary invalidity contentions are provided as an exhibit to Akamai's "Preliminary Invalidity Contentions" document dated June 8, 2016, and should be read in concert with the preliminary statement, reservation of rights, and additional invalidity grounds set forth therein. As set forth more fully therein, Akamai does not acquiesce to any particular interpretation or claim construction of any feature or limitation of the '155 patent, nor does Akamai concede enablement of any particular feature or limitation of the '155 patent under 35 U.S.C. § 112. Akamai reserves the right to argue its non-infringement and invalidity positions in the alternative. Certain prior art references may contain technical features that may be the same as or substantially similar to features contained within the Accused Products. Akamai asserts in its preliminary non-infringement contentions that the claim limitations in the '155 patent do not cover those features. However, to the extent the claims are construed to cover such technical features, the '155 patent is invalid based on the presence of those same technical features in the prior art references. To the extent Akamai's non-infringement contentions require certain claim constructions, such claim constructions should not be read into Akamai's invalidity contentions; similarly, to the extent Akamai's invalidity contentions require certain claim constructions, such claim constructions shall not be read into Akamai's non-infringement contentions.

[2] Chu is available as prior art against all claims of the '155 patent under pre-AIA 35 U.S.C. § 102(b).

[3] Devanneaux is available as prior art against all claims of the '155 patent under pre-AIA 35 U.S.C. § 102(b).

Asserted Claim	Disclosures in the Prior Art
	streaming video application on a fixed network transferring real-time video from a remote server."
establishing a first connection at the server for communicating with the end user computer;	"FIG. 1 illustrates two computer systems communicating over a network link 110. A sender application 104 in the sending computer system 102 uses a socket API 106 to pass data to a network stack 108, which packetizes the data and sends it over a network link 110 to a receiving computer system 112. The network stack 108 on the receiving computer system 112 processes the packets and passes them up to the receiving application 114 through the socket API 106." Chu at ¶ 31.  COMPUTER SYSTEM 102 COMPUTER SYSTEM 112 SENDER APPLICATION 104 SOCKET API 106 TCP LAYER IP LAYER LINK LAYER NETWORK STACK 108 RECEIVING APPLICATION 114 SOCKET API 106 TCP LAYER IP LAYER LINK LAYER NETWORK LINK 110

Asserted Claim	Disclosures in the Prior Art
receiving a request for content from the end user computer over the first connection, the request include a universal resource locator (URL);	"Depending on system policy, the appropriate technique for a connection may be chosen at a very fine granularity, and may vary dynamically, based on: user input or specification of priority; application input or preference; an application type; system policy; the source and/or destination port numbers used by the network connection; the source and/or destination Internet Protocol (IP) addresses of the network connection; the protocol used by the network connection; the characteristics of the network connection, including latency, bandwidth, loss-rate, and traffic characteristics; the service provided by the network connection; cached path characteristics from past connections; the location of the computer system and the second computer system; or any combination of the above. For instance, a connection to a local wireless IP address may need different TCP behavior than a streaming video application on a fixed network transferring real-time video from a remote server. The system can maintain a list of candidate functions for TCP behavior from which the application or user chooses, or in a further embodiment, privileged users can defined and plug-in their own functions, subject to a control policy that deters abusive network behavior." Chu at ¶¶ 54-66. "… TCP connections are established in order to use HyperText Transfer Protocol (HTTP) to communicate information requests from clients 102 to servers 206 and responses from servers 206 to clients 102." '155 patent, col. 6, ln 1-4 – HTTP is a common protocol used by Internet servers and clients, and Chu discloses, "The transmission control protocol (TCP) is part of the core Internet protocol which is used to transfer data between computing devices." Chu at ¶ 5.
determining one or more parameters relating to the performance of the first connection using information from the request, wherein the determined one or more parameters relate to utilization of available processing or memory capabilities of part or	"In a further variation, the set of functions are triggered by events that include: the receipt of a positive acknowledgement indicating that a packet was received; the receipt of negative acknowledgements indicating that packets may have been lost; the receipt of a selective acknowledgement that identifies received packets; the expiration of a timer; the elapse of a round-trip time interval; a call-back occurring either before or after a packet transmission; and the receipt of an explicit congestion notification (ECN). In a further variation, triggering an event prompts the set of functions to update the set of parameters." Chu at ¶¶ 14-22. This disclosure would have been sufficient to render any missing element obvious to a person of ordinary skill in the art at the relevant time. Devanneaux teaches: This baseline tag is used to temporarily stop prefetching if a given edge server CPU

Asserted Claim	Disclosures in the Prior Art

all of a system supporting the first connection;	utilization percent is above this threshold:
	`<edgeservices:prefetch.percent-threshold>90</edgeservices:prefetch.percent-threshold>`
	This is a baseline tag that defines a maximum number of objects to keep in the edge server cache for non-cacheable prefetched objects:
	`<edgeservices:prefetch.cache.max-objects>1000</edgeservices:prefetch.cache.max-objects>`
	Devanneaux at ¶¶ 79-80.
	
	"If desired, prefetching can be combined with other edge server features, such as path optimization, TCP connection optimization, content compression optimizations, and the like." Devanneaux at ¶ 83.

determining one or more first values of attributes based on the URL and the one or more parameters;	"One embodiment of the present invention provides network resource- and bandwidth-control by extending the plug-in architecture to allow different TCP behaviors to be plugged-in on a per-connection basis. The system maintains a vector of function pointers that point to the chosen TCP technique for each connection. Depending on system policy, the appropriate technique for a connection may be chosen at a very fine granularity, and may vary dynamically, based on: user input or specification of priority; application input or preference; an application type; system policy; the source and/or destination port numbers used by the network connection; the source and/or destination Internet Protocol (IP) addresses of the network connection; the protocol used by the network connection; the characteristics of the network connection, including latency, bandwidth, loss-rate, and traffic characteristics; the service provided by the network connection; cached path characteristics from past connections; the location of the computer system and the second computer system; or any combination of the above. For instance, a connection to a local wireless IP address may need different TCP behavior than a streaming video application on a fixed network transferring real-time video from a remote server. The system can maintain a list of candidate functions for TCP behavior from which the application or user chooses, or in a further embodiment, privileged users can defined and plug-in their own functions, subject to a control policy that deters abusive network behavior." Chu at ¶¶ 54-66.

Asserted Claim	Disclosures in the Prior Art
	"… TCP connections are established in order to use HyperText Transfer Protocol (HTTP) to communicate information requests from clients 102 to servers 206 and responses from servers 206 to clients 102." '155 patent, col. 6, ln 1-4 – HTTP is a common protocol used by Internet servers and clients, and Chu discloses, "The transmission control protocol (TCP) is part of the core Internet protocol which is used to transfer data between computing devices." Chu at ¶ 5.
modifying second values of attributes for the first connection at a transport layer to result in the determined one or more first values, the second values of the attributes for the first connection thereafter influencing utilization of the available processing or memory capabilities of the part or all of the system supporting the first connection;	"Per-Connection TCP Congestion Control. While a plug-in architecture for TCP allows TCP behavior to be changed at the system level, each network connection may encounter different conditions based on the destination or other factors, so a more ideal solution allows multiple techniques to be applied simultaneously on the computer system. One embodiment of the present invention provides network resource- and bandwidth-control by extending the plug-in architecture to allow different TCP behaviors to be plugged-in on a per-connection basis. The system maintains a vector of function pointers that point to the chosen TCP technique for each connection. Depending on system policy, the appropriate technique for a connection may be chosen at a very fine granularity, and may vary dynamically, based on: user input or specification of priority; application input or preference; an application type; system policy; the source and/or destination port numbers used by the network connection; the source and/or destination Internet Protocol (IP) addresses of the network connection; the protocol used by the network connection; the characteristics of the network connection, including latency, bandwidth, loss-rate, and traffic characteristics; the service provided by the network connection; cached path characteristics from past connections; the location of the computer system and the second computer system; or any combination of the above …" Chu at ¶¶ 53-66. "Fine-grained per-connection control of TCP behavior enables additional possibilities not available with a traditional hard-wired TCP layer. …A plug-in function for a connection can provide a level of QoS [quality-of-service] and bandwidth control directly inside the TCP layer, thereby taking advantage of knowledge that is difficult to obtain from outside the transport layer." Chu at ¶ 68. "A Plug-In Architecture for TCP Congestion Control. The present invention extends existing network stacks (including stacks deployed in kernel space, user space, and/or in TCP offload engines) to allow core functions of the TCP congestion control system to be changed easily and dynamically. While many portions of the TCP implementation contribute to TCP dynamics, only a subset of the implementation is

Asserted Claim	Disclosures in the Prior Art
	likely to still evolve. One such area still seeing significant changes is transmission-side congestion avoidance. In one embodiment of the present invention, a subset of the TCP transmit functionality becomes a swappable plug-in, while the standardized and unchanging portion of the TCP layer remains hard-wired. The system enters the swappable portion whenever an event is encountered that triggers a recomputation of congestion parameters, for instance cwnd, ssthresh, and RTT. Such triggers for the TCP sender side include: the receipt of new data to be sent; the receipt of a positive acknowledgement indicating that a packet was received; the receipt of negative acknowledgements indicating that packets may have been lost; the receipt of a selective acknowledgement that identifies a received packet; the expiration of a timer; the elapse of a round-trip time interval; a call-back occurring either before or after a packet transmission or re-transmission; and the receipt of an explicit congestion notification (ECN). The plug-in module includes a set of functions that are invoked in response to the above events. These functions can be given access to fields from the TCP layer, such as the TCP control block and headers of acknowledgement packets, thereby allowing the plug-in to work directly with the raw TCP parameters. Allowing this type of access, instead of creating an abstraction on top of TCP, enables all approaches of congestion avoidance, including loss-based and delay-based approaches. The main output from these functions is a set of recomputed parameters (e.g. cwnd, ssthresh, RTT), which are then fed back into the hard-wired portion of the TCP implementation to continue execution. Chu at ¶¶ 39-48. "…changing the set of functions changes the transmit and receive characteristics of the network stack, thereby changing the congestion-control technique for the network stack." Chu at ¶ 13.
changing, on a connection-specific basis, a connection protocol stack operator based upon the modified values of the attributes; and	"Per-Connection TCP Congestion Control. While a plug-in architecture for TCP allows TCP behavior to be changed at the system level, each network connection may encounter different conditions based on the destination or other factors, so a more ideal solution allows multiple techniques to be applied simultaneously on the computer system. One embodiment of the present invention provides network resource- and bandwidth-control by extending the plug-in architecture to allow different TCP behaviors to be plugged-in on a per-connection basis. The system maintains a vector of function pointers that point to the chosen TCP technique for each connection. Depending on system policy, the appropriate technique for a connection may be chosen at a very fine granularity, and may vary dynamically, based on: …" Chu at ¶¶ 53-54.

Asserted Claim	Disclosures in the Prior Art
	"FIG. 3 presents a flow chart illustrating the process of changing the TCP behavior of a network connection. The system first is notified of a need for changing the TCP behavior of a network connection (step 302). In response, the system disables a relevant portion of the network stack in order to put the network connection into a quiescent state (step 304). Then, the system changes the function pointer for the function associated with the TCP behavior to point to a new function with the desired behavior (step 306). Finally, the system re-enables the corresponding portion of the network stack to return the network connection to an active state (step 308)." Chu at ¶ 67. "One embodiment of the present invention provides a plug-in architecture for a network stack in an operating system. The network stack includes a set of functions configured to modify a set of parameters that are likely to change based on the network environment. The architecture includes a plug-in framework within the network stack that allows the set of functions to be dynamically changed in order to change the TCP behavior of the network stack to suit the network environment." Chu at ¶ 8. "In a further variation, triggering an event prompts the set of functions to update the set of parameters." Chu at ¶ 22. "In a variation on this embodiment, changing the set of functions allows the network stack to dynamically change TCP behavior and thereby transmit efficiently across diverse and changing network environments." Chu at ¶ 25. "The plug-in architecture allows the system to switch between different congestion avoidance techniques. Each technique uses a different approach, and may therefore maintain a different set of internal state. For instance, a delay-based technique such as Fast-TCP may track average queuing delay as well as minimum and biased RTTs, while TCP-Westwood gleans data from successive acknowledgment packets to compute an eligible rate estimate (ERE). Alternatively, High-Speed TCP (HS-TCP), a loss-based technique, keeps an internal table of congestion window sizes (i.e. a table for 'a(cwnd)' and 'b(cwnd)'). … The system can effectively give full control of TCP behavior to the plug-in by only allowing control parameters to be changed in the plugged-in functions." Chu at ¶ 50. "Allowing the TCP behavior to be easily modified, either manually or dynamically, provides an

Asserted Claim	Disclosures in the Prior Art
	opportunity to tune network performance of production networks as well as provide a flexible way to explore, implement, and test new congestion control techniques." Chu at ¶ 51.
sending the requested content from the server to the end user computer such that the transport layer manages delivery of the content in accordance with the modified second values of the attributes.	"FIG. 3 presents a flow chart illustrating the process of changing the TCP behavior of a network connection. The system first is notified of a need for changing the TCP behavior of a network connection (step 302). In response, the system disables a relevant portion of the network stack in order to put the network connection into a quiescent state (step 304). Then, the system changes the function pointer for the function associated with the TCP behavior to point to a new function with the desired behavior (step 306). Finally, the system re-enables the corresponding portion of the network stack to return the network connection to an active state (step 308)." Chu at ¶ 67. *See also,* Chu at ¶¶ 53-66.
2. The method of claim 1, wherein the one or more first values of attributes are further based on an estimated location of the end user computer.	"Per-Connection TCP Congestion Control. While a plug-in architecture for TCP allows TCP behavior to be changed at the system level, each network connection may encounter different conditions based on the destination or other factors, so a more ideal solution allows multiple techniques to be applied simultaneously on the computer system. One embodiment of the present invention provides network resource- and bandwidth-control by extending the plug-in architecture to allow different TCP behaviors to be plugged-in on a per-connection basis. The system maintains a vector of function pointers that point to the chosen TCP technique for each connection. Depending on system policy, the appropriate technique for a connection may be chosen at a very fine granularity, and may vary dynamically, based on: user input or specification of priority; application input or preference; an application type; system policy; the source and/or destination port numbers used by the network connection; the source and/or destination Internet Protocol (IP) addresses of the network connection; the protocol used by the network connection; the characteristics of the network connection, including latency, bandwidth, loss-rate, and traffic characteristics; the service provided by the network connection; cached path characteristics from past connections; the location of the computer system and the second computer system; or any combination of the above …" Chu at ¶¶ 53-66.
3. The method of claim 1, further comprising determining a latency characteristic of the	"One embodiment of the present invention provides network resource- and bandwidth-control by extending the plug-in architecture to allow different TCP behaviors to be plugged-in on a per-connection basis. The system maintains a vector of function pointers that point to the chosen TCP technique for each connection. Depending on system policy, the appropriate technique for a connection may be chosen at a very fine

Asserted Claim	Disclosures in the Prior Art
first connection, wherein at least one of the second values of the attributes for the first connection is modified based on the latency characteristic.	granularity, and vary dynamically, based on: … the characteristics of the network connection, including latency, bandwidth, loss-rate, and traffic characteristics; …" Chu at ¶¶ 54-62. "The main output from these functions is a set of recomputed parameters (e.g. cwnd, ssthresh, RTT), which are then fed back into the hard-wired portion of the TCP implementation to continue execution." Chu at ¶ 48. "…changing the set of functions changes the transmit and receive characteristics of the network stack, thereby changing the congestion-control technique for the network stack." Chu at ¶ 13. *See also* Chu at ¶¶ 39-48 regarding modifying values of attributes.
4. The method of claim 1, further comprising: determining a connection type of the end user computer; and determining a latency characteristic associated with the connection type, wherein at least one of the second values of the attributes for the first connection is modified based on the latency characteristic associated with the connection type of the end user computer.	"Per-Connection TCP Congestion Control. While a plug-in architecture for TCP allows TCP behavior to be changed at the system level, each network connection may encounter different conditions based on the destination or other factors, so a more ideal solution allows multiple techniques to be applied simultaneously on the computer system. One embodiment of the present invention provides network resource- and bandwidth-control by extending the plug-in architecture to allow different TCP behaviors to be plugged-in on a per-connection basis. The system maintains a vector of function pointers that point to the chosen TCP technique for each connection. Depending on system policy, the appropriate technique for a connection may be chosen at a very fine granularity, and may vary dynamically, based on: user input or specification of priority; application input or preference; an application type; system policy; the source and/or destination port numbers used by the network connection; the source and/or destination Internet Protocol (IP) addresses of the network connection; the protocol used by the network connection; the characteristics of the network connection, including latency, bandwidth, loss-rate, and traffic characteristics; the service provided by the network connection; cached path characteristics from past connections; the location of the computer system and the second computer system; or any combination of the above …" Chu at ¶¶ 53-66.
6. The method of claim	"The plug-in approach also enables employing an aggressive, special-purpose technique in a controlled

Asserted Claim	Disclosures in the Prior Art
1, further comprising determining an autonomous system from which the first connection is received, wherein at least one of the second values of the attributes for the first connection is modified based on network characteristics of the autonomous system.	network environment. For instance, a server in a data center with a well-controlled traffic pattern or well-tuned queuing model might deploy a non-compliant congestion control technique that allows packets to be sent without slow-start or any bandwidth throttling. This technique could be useful, for example, to eliminate the overhead of congestion control for connections that transfer data between two servers on a dedicated network link, or to expedite connections that exchange cluster membership heartbeat messages within the data center. Previously, such service variation either was not possible, or would require multiple servers." Chu at ¶ 69.
7. The method of claim 1, further comprising determining a link utilization between the server and an autonomous system of the end user computer, wherein at least one of the second values of the attributes for the first connection is modified based on the link utilization.	"The plug-in approach also enables employing an aggressive, special-purpose technique in a controlled network environment. For instance, a server in a data center with a well-controlled traffic pattern or well-tuned queuing model might deploy a non-compliant congestion control technique that allows packets to be sent without slow-start or any bandwidth throttling. This technique could be useful, for example, to eliminate the overhead of congestion control for connections that transfer data between two servers on a dedicated network link, or to expedite connections that exchange cluster membership heartbeat messages within the data center. Previously, such service variation either was not possible, or would require multiple servers." Chu at ¶ 69
8. The method of claim 1, further comprising determining a predetermined performance profile for	"One embodiment of the present invention provides network resource- and bandwidth-control by extending the plug-in architecture to allow different TCP behaviors to be plugged-in on a per-connection basis. The system maintains a vector of function pointers that point to the chosen TCP technique for each connection. Depending on system policy, the appropriate technique for a connection may be chosen at a very fine granularity, and vary dynamically, based on: user input or specification of priority; application input or

Asserted Claim	Disclosures in the Prior Art
the first connection using the information from the request, wherein at least one of the second values of the attributes for the first connection is modified based on the predetermined performance profile.	preference; an application type; system policy; the source and/or destination port numbers used by the network connection; the source and/or destination Internet Protocol (IP) addresses of the network connection; the protocol used by the network connection; the characteristics of the network connection, including latency, bandwidth, loss-rate, and traffic characteristics; the service provided by the network connection; cached path characteristics from past connections; the location of the computer system and the second computer system; or any combination of the above …" Chu at ¶¶ 54-66. "The main output from these functions is a set of recomputed parameters (e.g. cwnd, ssthresh, RTT), which are then fed back into the hard-wired portion of the TCP implementation to continue execution." Chu at ¶ 48. "…changing the set of functions changes the transmit and receive characteristics of the network stack, thereby changing the congestion-control technique for the network stack." Chu at ¶ 13. See also Chu at ¶¶ 39-48 regarding modifying values of attributes.
10. The method of claim 1, wherein modifying the second values of the attributes for the first connection thereafter adjusts a timing of data transmission at the transport layer in accordance with the one or more parameters.	"The TCP layer comprises an important part of the network stack 108. The core of the TCP protocol is based on a set of parameters that together determine a set of data packets, a timeframe in which they will be transmitted from the sender side, and how acknowledgements will be generated on the receiving side. The sending side constantly recalculates the set of parameters based on feedback from, for instance, acknowledgement packets and local timers, in order to decide which data to send or resend, and when. Important parameters include: 'RTT', the round-trip time it takes a data packet to travel from the sender to the receiver; 'cwnd,' the size of the congestion window, which specifies the number of data packets that can be transmitted without having received corresponding acknowledgement packets; and 'ssthresh,' the slow-start threshold, which determines how the size of the congestion window increases. The receiver side, meanwhile, decides when to generate either positive, negative, or selective acknowledgements." Chu at ¶¶ 32-35. *See also* Chu at ¶¶13, 39-48 regarding modifying values of attributes.
11. The method of	"The TCP layer comprises an important part of the network stack 108. The core of the TCP protocol is

Asserted Claim	Disclosures in the Prior Art
claim 1, wherein modifying the second values of the attributes for the first connection thereafter adjusts a transport layer send window associated with the first connection.	based on a set of parameters that together determine a set of data packets, a timeframe in which they will be transmitted from the sender side, and how acknowledgements will be generated on the receiving side. The sending side constantly recalculates the set of parameters based on feedback from, for instance, acknowledgement packets and local timers, in order to decide which data to send or resend, and when. Important parameters include: 'RTT', the round-trip time it takes a data packet to travel from the sender to the receiver; 'cwnd,' the size of the congestion window, which specifies the number of data packets that can be transmitted without having received corresponding acknowledgement packets; and 'ssthresh,' the slow-start threshold, which determines how the size of the congestion window increases. The receiver side, meanwhile, decides when to generate either positive, negative, or selective acknowledgements." Chu at ¶¶ 32-35. *See also* Chu at ¶¶ 13, 39-48 regarding modifying values of attributes.
13. A content distribution server comprising:	"The transmission control protocol (TCP) is part of the core Internet protocol which is used to transfer data between computing devices. … [ ] what is needed are architectures and methods that facilitate congestion control for TCP without the limitations of existing approaches. One embodiment of the present invention provides a plug-in architecture for a network stack in an operating system. The network stack includes a set of functions configured to modify a set of parameters that are likely to change based on the network environment. The architecture includes a plug-in framework within the network stack that allows the set of functions to be dynamically changed in order to change the TCP behavior of the network stack to suit the network environment." Chu at ¶¶ 5-8. Examples of distribution of content are presented in Chu, e.g., at ¶ 66: "For instance, a connection to a local wireless IP address may need different TCP behavior than a streaming video application on a fixed network transferring real-time video from a remote server." Chu indicates that architecture can be/include a server. *See e.g.*, Chu at ¶¶ 69-70.
a network interface having a plurality of ports configured to send and receive data	*See* the "Network Link 110" as shown in FIG. 1. "FIG. 1 illustrates two computer systems communicating over a network link 110. A sender application 104 in the sending computer system 102 uses a socket API 106 to pass data to a network stack 108, which

Asserted Claim	Disclosures in the Prior Art
over a connecting network;	packetizes the data and sends it over a network link 110 to a receiving computer system 112. The network stack 108 on the receiving computer system 112 processes the packets and passes them up to the receiving application 114 through the socket API 106." Chu at ¶ 31.
a processor coupled to the network interface and configured to manage a plurality of connections to end user computers;	"The transmission control protocol (TCP) is part of the core Internet protocol which is used to transfer data between computing devices." Chu at ¶ 5. *See also* Chu at FIG. 1, "TCP Layer" of the "Network Stack 108" of "Computer System 102". *See e.g.*, FIG. 2, "TCP transmit processing 202" and "TCP receive processing 204" Chu at ¶ 49. "…the source and/or destination Internet Protocol (IP) addresses of the network connection …" Chu at ¶ 60 (multiple recipients)
a protocol handler configured to establish the plurality of connections with the end user computers according to predetermined transport layer parameters of the content distribution server and to manage data transmission over the plurality of connections; and	"One embodiment of the present invention provides network resource- and bandwidth-control by extending the plug-in architecture to allow different TCP behaviors to be plugged-in on a per-connection basis." Chu at ¶ 54. "A plug-in function for a connection can provide a level of QoS and bandwidth control directly inside the TCP layer [*inside the transport layer*], thereby taking advantage of knowledge that is difficult to obtain from outside of the transport layer. For instance, in a traditional system, an attempt to throttle-down transmission might be interpreted as a sign of congestion and/or time-out, and prompt undesired re-transmission. The traditional approach of performing resource control and bandwidth management outside of the transport layer at a fine granularity also incurs heavy processing overhead in parsing headers and maintaining state on a per-flow basis. In the present invention, such capabilities can be added to the TCP behavior using a plug-in and handled appropriately." Chu at ¶ 68.
a data source configured to supply requested content to the end user computers over the plurality of connections, wherein	Chu at FIG. 1 "[sending] Computer System 102" "FIG. 1 illustrates two computer systems communicating over a network link 110. A sender application 104 in the sending computer system 102 uses a socket API 106 to pass data to a network stack 108, which packetizes the data and sends it over a network link 110 to a receiving computer system 112. The network stack 108 on the receiving computer system 112 processes the packets and passes them up to the receiving

Asserted Claim	Disclosures in the Prior Art
the data source is configured to monitor a first connection for a request to:	application 114 through the socket API 106." Chu at ¶ 31.
determine one or more parameters for the first connection based on the request, the determined one or more transport layer parameters relating to utilization of available processing or memory capabilities of part or all of a system supporting the first connection;	*See* citations above for the following limitations of claim 1: "determining one or more parameters relating to the performance of the first connection using information from the request, wherein the determined one or more parameters relate to utilization of available processing or memory capabilities of part or all of a system supporting the first connection."
determine one or more first values of attributes based on a URL and the one or more parameters, the request including the URL;	*See* citations above for the following limitations of claim 1: "determining one or more first values of attributes based on the URL and the one or more parameters."
direct the protocol handler to modify second values of attributes for the first connection to result in the determined one or	*See* citations above for the following limitations of claim 1: "modifying second values of attributes for the first connection at a transport layer to result in the determined one or more first values, the second values of the attributes for the first connection thereafter influencing utilization of the available processing or memory capabilities of the part or all of the system supporting the first connection."

Asserted Claim	Disclosures in the Prior Art
more first values, the second values of the attributes for the first connection thereafter influencing utilization of the available processing or memory capabilities of the part or all of the system supporting the first connection; and	*See also*, regarding "protocol handler": "One embodiment of the present invention provides network resource- and bandwidth-control by extending the plug-in architecture to allow different TCP behaviors to be plugged-in on a per-connection basis." Chu at ¶ 54.
change a connection protocol stack operator based upon the modified second values of the attributes.	*See* citations above for the following limitations of claim 1: "changing, on a connection-specific basis, a connection protocol stack operator based upon the modified values of the attributes."
14. The content distribution server of claim 13, wherein the one or more first values of attributes are further based on an estimated location of the end user computer.	"Per-Connection TCP Congestion Control. While a plug-in architecture for TCP allows TCP behavior to be changed at the system level, each network connection may encounter different conditions based on the destination or other factors, so a more ideal solution allows multiple techniques to be applied simultaneously on the computer system. One embodiment of the present invention provides network resource- and bandwidth-control by extending the plug-in architecture to allow different TCP behaviors to be plugged-in on a per-connection basis. The system maintains a vector of function pointers that point to the chosen TCP technique for each connection. Depending on system policy, the appropriate technique for a connection may be chosen at a very fine granularity, and may vary dynamically, based on: user input or specification of priority; application input or preference; an application type; system policy; the source and/or destination port numbers used by the network connection; the source and/or destination Internet Protocol (IP) addresses of the network connection; the protocol used by the network connection; the characteristics of the network connection, including latency, bandwidth, loss-rate, and traffic characteristics; the service provided by the network connection; cached path characteristics from past connections; the location of the computer system and the second computer system; or any combination of

Asserted Claim	Disclosures in the Prior Art
	the above …" Chu at ¶¶ 53-66.
15. The content distribution server of claim 13, further comprising a protocol attribute information store having information relating data associated with a content request to one or more connection parameters, wherein the data source determines at least one of the first values based on information retrieved from the protocol attribute information store.	"One embodiment of the present invention provides network resource- and bandwidth-control by extending the plug-in architecture to allow different TCP behaviors to be plugged-in on a per-connection basis. The system maintains a vector of function pointers that point to the chosen TCP technique for each connection. Depending on system policy, the appropriate technique for a connection may be chosen at a very fine granularity, and may vary dynamically, based on: user input or specification of priority; application input or preference; an application type; system policy; the source and/or destination port numbers used by the network connection; the source and/or destination Internet Protocol (IP) addresses of the network connection; the protocol used by the network connection; the characteristics of the network connection, including latency, bandwidth, loss-rate, and traffic characteristics; the service provided by the network connection; cached path characteristics from past connections; the location of the computer system and the second computer system; or any combination of the above. For instance, a connection to a local wireless IP address may need different TCP behavior than a streaming video application on a fixed network transferring real-time video from a remote server. The system can maintain a list of candidate functions for TCP behavior from which the application or user chooses, or in a further embodiment, privileged users can defined and plug-in their own functions, subject to a control policy that deters abusive network behavior." Chu at ¶¶ 54-66. *See also* Chu at ¶¶ 8, 22, 25, 50, 51, and 67.
16. The content distribution server of claim 13, wherein the data source determines a geographic region corresponding to a destination address of the first connection, and wherein at least one of the one or more first values is further based	"Per-Connection TCP Congestion Control. While a plug-in architecture for TCP allows TCP behavior to be changed at the system level, each network connection may encounter different conditions based on the destination or other factors, so a more ideal solution allows multiple techniques to be applied simultaneously on the computer system. One embodiment of the present invention provides network resource- and bandwidth-control by extending the plug-in architecture to allow different TCP behaviors to be plugged-in on a per-connection basis. The system maintains a vector of function pointers that point to the chosen TCP technique for each connection. Depending on system policy, the appropriate technique for a connection may be chosen at a very fine granularity, and may vary dynamically, based on: user input or specification of priority; application input or preference; an application type; system policy; the source and/or destination port numbers used by the network connection; the source and/or destination Internet Protocol (IP) addresses of the network connection; the protocol used by the network connection; the

Asserted Claim	Disclosures in the Prior Art
on the geographic region.	characteristics of the network connection, including latency, bandwidth, loss-rate, and traffic characteristics; the service provided by the network connection; cached path characteristics from past connections; the location of the computer system and the second computer system; or any combination of the above …" Chu at ¶¶ 53-66.
17. The content distribution server of claim 13, wherein the data source determines an autonomous system associated with the first connection, and wherein at least one of the one or more first values is further based on network characteristics of the autonomous system.	The plug-in approach also enables employing an aggressive, special-purpose technique in a controlled network environment. For instance, a server in a data center with a well-controlled traffic pattern or well-tuned queuing model might deploy a non-compliant congestion control technique that allows packets to be sent without slow-start or any bandwidth throttling. This technique could be useful, for example, to eliminate the overhead of congestion control for connections that transfer data between two servers on a dedicated network link, or to expedite connections that exchange cluster membership heartbeat messages within the data center. Previously, such service variation either was not possible, or would require multiple servers. Chu at ¶ 69
18. The content distribution server of claim 13, wherein the data source determines a latency characteristic associated with the first connection based on the information from the request and directs the protocol attribute selector to modify at least one of the second values of the attributes	*See* citations above for the following limitations of claim 3: "determining a latency characteristic of the first connection, wherein at least one of the second values of the attributes for the first connection is modified based on the latency characteristic."

Asserted Claim	Disclosures in the Prior Art
based on the latency characteristic.	
19. The content distribution server of claim 13, wherein the protocol handler is configured to adjust a timing of data transmission at a transport layer based on the one or more first values.	*See* citations above for the following limitations of claim 10: "wherein modifying the second values of the attributes for the first connection thereafter adjusts a timing of data transmission at the transport layer in accordance with the one or more parameters."
20. A computer program product comprising a non-transitory computer-readable medium encoded with one or more sequences of one or more instructions which, when executed by a processor, cause a computer to:	"One embodiment of the present invention provides a plug-in architecture for a network stack in an operating system. The network stack includes a set of functions configured to modify a set of parameters that are likely to change based on the network environment. The architecture includes a plug-in framework within the network stack that allows the set of functions to be dynamically changed in order to change the TCP behavior of the network stack to suit the network environment." Chu at Abstract
establishing a first connection at the server for communicating with an end user computer;	*See* citations above for the following limitations of claim 1: "establishing a first connection at the server for communicating with the end user computer."
receiving a request for content from the end	*See* citations above for the following limitations of claim 1:

Asserted Claim	Disclosures in the Prior Art
user computer over the first connection, the request include a universal resource locator (URL);	"receiving a request for content from the end user computer over the first connection, the request include a universal resource locator (URL)."
determining one or more parameters relating to the performance of the first connection using information from the request, wherein the determined one or more parameters relate to utilization of available processing or memory capabilities of part or all of a system supporting the first connection;	*See* citations above for the following limitations of claim 1: "determining one or more parameters relating to the performance of the first connection using information from the request, wherein the determined one or more parameters relate to utilization of available processing or memory capabilities of part or all of a system supporting the first connection."
determining one or more first values of attributes based on the URL and the one or more parameters;	*See* citations above for the following limitations of claim 1: "determining one or more first values of attributes based on the URL and the one or more parameters."
modifying second values of attributes for the first connection at a transport layer to result in the determined one	*See* citations above for the following limitations of claim 1: "modifying second values of attributes for the first connection at a transport layer to result in the determined one or more first values, the second values of the attributes for the first connection thereafter influencing utilization of the available processing or memory capabilities of the part or all of the system

Asserted Claim	Disclosures in the Prior Art
or more first values, the second values of the attributes for the first connection thereafter influencing utilization of the available processing or memory capabilities of the part or all of the system supporting the first connection;	supporting the first connection."
changing, on a connection-specific basis, a connection protocol stack operator based upon the modified values of the attributes ; and	*See* citations above for the following limitations of claim 1: "changing, on a connection-specific basis, a connection protocol stack operator based upon the modified values of the attributes."
sending the requested content from the server to the end user computer such that the transport layer manages delivery of the content in accordance with the modified second values of the attributes.	*See* citations above for the following limitations of claim 1: "sending the requested content from the server to the end user computer such that the transport layer manages delivery of the content in accordance with the modified second values of the attributes."